UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/924,335	08/07/2001	Daniel Leibholz	P4806/SMQ-021	2997

959 7590 01/04/2005

LAHIVE & COCKFIELD, LLP.
28 STATE STREET
BOSTON, MA 02109

EXAMINER	
GERSTL, SHANE F	
ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 01/04/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/924,335	LEIBHOLZ ET AL.	
	Examiner	Art Unit	
	Shane F Gerstl	2183	

-- The MAILING DATE of this communication appears n the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *15 September 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-16 and 18-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-16 and 18-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *29 March 2002* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-20 have been examined.

Papers Received

2. Receipt is acknowledged of the amendment papers where the papers have been placed of record in the file.

Specification

3. The title of the invention is not descriptive. As shown in MPEP 606, the title needs to be clearly indicative of the invention. The original and amended titles are not clearly indicative of the invention but only of the general art the invention is a member of (out-of-order simultaneous multithreaded processors). A new title is required that is clearly indicative of the invention to which the claims are directed.

The following title is suggested: IN A MULTI-THREADING MODE GIVING EACH THREAD EXCLUSIVE ACCESS TO ONE REGISTER FILE AND IN A SINGLE THREAD MODE GIVING THE ACTIVE THREAD ACCESS TO MULTIPLE REGISTER FILES.

Claim Objections

4. Claim 1 is objected to because of the following informalities: the second to last line is missing a word to make the limitation grammatically correct. The Examiner is taking this line to read "the second register file where a first thread has access to the first register file and a" with the word "where" inserted. Appropriate correction is required.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

6. Claims 9 and 10 are rejected under 35 U.S.C. 102(b) as being anticipated by Levy (6,092,175).

7. In regard to claim 9, Levy discloses in a microprocessor having a first register file and a second register file, a mapping mechanism for mapping references to registers in both the first register file and the second register file when in a single thread mode and for mapping references to registers in instructions of a first thread solely to registers in the first register file and references to registers in instructions in a second thread to registers in the second register file when in a multi-thread mode. [The Examiner notes that the claim language does not require in the single thread mode the mapping of a single register name to a register in both the first and second register files, but instead simply states mapping references to registers in both the first and second register file with no mention of mapping to a single name to both registers as given in the other independent claims. Thus, Levy does anticipate claim 9 since he uses register files of inactive threads (including when there is only one active thread – a single thread mode) for extra physical registers (as indicated by Applicant). Column 5, lines 21-42 and figure 5C discuss an embodiment with the use of privately used architectural registers for each thread. This means that a first thread has a first set of architectural registers and the

second thread has a second set of architectural registers. Column 10, lines 18-41 show that this third embodiment is called SSASR and provides further details of it. Column 15, lines 28-41 show that the architectural registers are set up in eight separate register files, one for each of eight threads (column 13, line 1). Therefore the private architectural register sets are in fact architecture register files. Column 5, lines 21-42 show that an embodiment has private architectural registers for each thread in a simultaneous multithreaded processor (SMT, column 4, line 66 – column 5, line 1) for a multi-thread mode, when the maximum number of threads are executing, where there is no register sharing. This section also shows that at some points in time, the processor will be executing fewer threads than the number of private register sets (files) contained in the processor. This number of threads fewer than the maximum number of threads (eight) includes the case of when a single thread is executing, or a single thread mode. Further proof of this is shown in figures 6A-6D, for example, where it is shown that SSASR implementations (different only by register length) all execute a single thread at times. Column 5, lines 21-42 show that in the SSASR embodiment when fewer threads than the maximum are executed, and thus the single thread mode as well, the architectural registers (and thus register files) of the non-executing threads or contexts are shared with the active threads. Therefore, when there is a single executing thread (single thread mode) architectural registers will be used from the second thread, which is idle, and thus in a single thread mode the active thread has access to both the first and second register files. As shown above the register files are architectural and thus the instructions register references are directly mapped to the registers in these files.

For the multi-thread mode, each thread only access registers private to that thread (first thread/first register file) and for the single thread mode both first and second register files are used with the second register file being used as rename registers for mapping of the architectural registers of the first register file. Column 5, lines 21-42 show that architectural registers files from the idle threads (i.e. second register file from the second thread) are used for renaming registers and thus are mapped to the architectural registers of the first register file.]

8. In regard to claim 10, Levy discloses the mapping mechanism of claim 9, wherein the mapping mechanism switches between the single thread mode and the multi-thread mode when directed by a control mechanism. [As shown above when there is one thread active the processor is in single thread mode and when all threads are active the processor is in multi-thread mode, and thus there must be a mechanism to switch the register access properties of the processor for switching between these modes. This mechanism is element 28 of figure 1, which is described in column 7, lines 18-22 and column 10, lines 34-37 as deciding what registers to use for holding data for each thread.]

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 1-3, 6-8, 11-13, 15, 16, and 18-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Levy in view of Cook (5,301,340).

11. In regard to claim 1,

a. Levy discloses a microprocessor, comprising:

i. a first register file containing registers;

ii. a second register file containing registers; [Column 5, lines 21-42 and figure 5C discuss a third embodiment of Levy with the use of privately used architectural registers for each thread. This means that a first thread has a first set of architectural registers and the second thread has a second set of architectural registers. Column 10, lines 18-41 show that this third embodiment is called SSASR and provides further details of it. Column 15, lines 28-41 show that the architectural registers are set up in eight separate register files, one for each of eight threads (column 13, line 1) and thus a first and second register file containing registers are disclosed.]

iii. and a facility for granting access to the first and second register files in a single thread mode and in a multi-thread mode, wherein, in the single thread mode, a single thread has access to both the first register file and the second register file and wherein, in the multi-thread mode, where the register name refers to one location in one of the first register file and the second register file a first thread has access to the first register file and a second thread has access to the second register file. [Column 5, lines

21-42 show that an embodiment has private architectural registers for each thread in a simultaneous multithreaded processor (SMT, column 4, line 66 – column 5, line 1) for a multi-thread mode, when the maximum number of threads are executing, where there is no register sharing. This section also shows that at some points in time, the processor will be executing fewer threads than the number of private register sets (files) contained in the processor. This number of threads fewer than the maximum number of threads (eight) includes the case of when a single thread is executing, or a single thread mode. Further proof of this is shown in figures 6A-6D, for example, where it is shown that SSASR implementations (different only by register length) all execute a single thread at times. Column 5, lines 21-42 show that in the SSASR embodiment when fewer threads than the maximum are executed, and thus the single thread mode as well, the architectural registers (and thus register files) of the non-executing threads or contexts are shared with the active threads. Therefore, when there is a single executing thread (single thread mode) architectural registers will be used from the second thread, which is idle, and thus in a single thread mode the active thread has access to both the first and second register files. This all inherently occurs in some logic group or a facility. Figure 5C and the sections cited above show that the register files are used for renaming and thus in the multi-

thread mode each thread and the functional unit therein (see figures 1, 3, and 4) uses its own private register file (first and second register files).]

b. Levy does not explicitly disclose in the single thread mode that a register name is renamed to refer to a first location in the first register file and to a first location in the second register file. However, Levy does disclose the use of register renaming for the SSASR embodiment given above (figure 5C for example) for multiple functional units (figures 1, 3, and 4).

c. Cook has disclosed in figure 5 and column 4, lines 37-68 the idea of broadcasting the data to be written to each register file to all the shared register files to create identical register files or in effect a multi-port register file.

d. In column 5, lines 21-42 along with the background, for example, Levy has recognized that unused portions of the processor when less than the maximum number of threads are executed (including the single thread mode) is a waste of resources and thus proposes sharing these otherwise thread-private resources. Specifically Levy has taught this idea for the use of sharing private register files for renaming, but one of ordinary skill in the art would also have recognized the waste of resources the idle execution units provide and would have been motivated to use all of the execution units of Levy to optimize execution of the single thread in a superscalar manner when a single thread is executing in order to boost performance of that single thread. Column 2, lines 29-40 of Cook along with the section of Cook cited above show that by broadcasting the data to each private yet shared register file, in effect, a fast multi-port type register file is

created that allows for quick data transfers to and from the private register file to each corresponding execution unit. This quick performance would have motivated one of ordinary skill in the art to modify the design of Levy to broadcast the writes to each shared register file, when in single thread mode where the resources are shared.

It would have been obvious to one of ordinary skill in the art to modify the design of Levy to use idle execution units in a superscalar manner when in a single thread mode and to broadcast writes to each register file associated with an execution unit in order to keep identical copies so that increased performance is realized in the system both through superscalar execution and a multi-port nearby register file when a single thread is executed.

12. In regard to claim 2, Levy discloses the microprocessor of claim 1, wherein, in the single thread mode, the single thread has exclusive access to both the first register file and the second register file, relative to other threads. [As shown above, in single thread mode, only one thread is active and thus this is the only thread with access (exclusive access) to both register files since the other threads are idle.]

13. In regard to claim 3, Levy discloses the microprocessor of claim 1, wherein the facility includes a mechanism for switching between the single thread mode and the multi-thread mode. [As shown above when there is one thread active the processor is in single thread mode and when all threads are active the processor is in multi-thread mode, and thus there must be a mechanism to switch the register access properties of the processor for switching between these modes. This mechanism is element 28 of

figure 1, which is described in column 7, lines 18-22 and column 10, lines 34-37 as deciding what registers to use for holding data for each thread.]

14. In regard to claim 6, Levy discloses the microprocessor of claim 1, wherein the facility is a register renamer that maps logical register references in instructions to registers in the first register file and the second register file. [As shown above, the architectural registers files are used for renaming registers and thus are mapped to the logical registers of the instructions. Therefore, the facility inherently includes a register renamer for mapping the instruction references to the architectural register of the first register file and then to the rename registers of the second register file.]

15. In regard to claim 7, Levy discloses the microprocessor of claim 1, further comprising write lines extending between the first register file and the second register file so that contents of one of the first and second register files may be copied to the other of the first and second register files. [As shown in figure 5 of Cook, write lines connect all the register files in order to copy the contents being written to one register file to all the register files.]

16. In regard to claim 8, Levy discloses the microprocessor of claim 1, wherein the microprocessor supports simultaneous multi-threading (SMT), as shown above.

17. In regard to claim 11,

a. Levy discloses in a microprocessor having multiple register sets, a method, comprising the steps of:

i. providing respective threads that are simultaneously executing in a multi-thread mode with access to separate respective ones of the register sets;

ii. and switching from the multi-thread mode to a single thread mode where a single thread is executing.

[Column 5, lines 21-42 and figure 5C discuss an embodiment with the use of privately used architectural registers for each thread. This means that a first thread has a first set of architectural registers and the second thread has a second set of architectural registers. Column 10, lines 18-41 show that this third embodiment is called SSASR and provides further details of it. Column 5, lines 21-42 show that an embodiment has private architectural registers for each thread in a simultaneous multithreaded processor (SMT, column 4, line 66 – column 5, line 1) for a multi-thread mode, when the maximum number of threads are executing, where there is no register sharing. This section also shows that at some points in time, the processor will be executing fewer threads than the number of private register sets contained in the processor. This number of threads fewer than the maximum number of threads (eight) includes the case of when a single thread is executing, or a single thread mode. Further proof of this is shown in figures 6A-6D, for example, where it is shown that SSASR implementations (different only by register length) all execute a single thread at times. When there is one thread active the processor is in single thread mode and when all threads are active the processor is in multi-thread mode, and thus

there must be a mechanism to switch the register access properties of the processor for switching between these modes. This mechanism is element 28 of figure 1, which is described in column 7, lines 18-22 and column 10, lines 34-37 as deciding what registers to use for holding data for each thread.

b. Levy does not explicitly disclose where in the single thread mode, a single register name is renamed to refer to a location in each respective ones of the register sets. However, Levy does disclose the use of register renaming for the SSASR embodiment given above (figure 5C for example) for multiple functional units (figures 1, 3, and 4).

c. Cook has disclosed in figure 5 and column 4, lines 37-68 the idea of broadcasting the data to be written to each register file to all the shared register files to create identical register files or in effect a multi-port register file.

d. In column 5, lines 21-42 along with the background, for example, Levy has recognized that unused portions of the processor when less than the maximum number of threads are executed (including the single thread mode) is a waste of resources and thus proposes sharing these otherwise thread-private resources. Specifically Levy has taught this idea for the use of sharing private register files for renaming, but one of ordinary skill in the art would also have recognized the waste of resources the idle execution units provide and would have been motivated to use all of the execution units of Levy to optimize execution of the single thread in a superscalar manner when a single thread is executing in order to boost performance of that single thread. Column 2, lines 29-40 of Cook along

with the section of Cook cited above show that by broadcasting the data to each private yet shared register file, in effect, a fast multi-port type register file is created that allows for quick data transfers to and from the private register file to each corresponding execution unit. This quick performance would have motivated one of ordinary skill in the art to modify the design of Levy to broadcast the writes to each shared register file, when in single thread mode where the resources are shared.

It would have been obvious to one of ordinary skill in the art to modify the design of Levy to use idle execution units in a superscalar manner when in a single thread mode and to broadcast writes to each register file associated with an execution unit in order to keep identical copies so that increased performance is realized in the system both through superscalar execution and a multi-port nearby register file when a single thread is executed.

18. In regard to claim 12, Levy discloses the method of claim 11, wherein each of the register sets is a respective register file. [Column 15, lines 28-41 show that the architectural registers are set up in eight separate register files, one for each of eight threads (column 13, line 1). Therefore the private architectural register sets are in fact architecture register files.]

19. In regard to claim 13, Levy discloses the method of claim 11, where two simultaneously executing threads are provided in the multi-thread mode. [As shown above by using SMT.]

20. In regard to claim 15, Levy discloses the method of claim 11, wherein the switching comprises propagating values held in a first of the register sets to a second of the register sets to regain coherence between the first and second register sets. [As shown in figure 5 of Cook, write lines connect all the register files in order to propagate the contents being written to one register file to all the register files in order to regain coherence.]

21. In regard to claim 16,

a. Levy discloses a microprocessor, comprising:

i. a first bank of execution units for executing instructions;

ii. a second bank of execution units for executing instructions;

iii. a first register file having read and write ports associated with the first bank of execution units and read and write ports associated with the second bank of execution units;

iv. a second register file having read and write ports associated with the first bank of execution units and read and write ports associated with the second bank of execution units;

v. and circuitry for enabling or disabling selected ones of the read ports and the write ports to control access by threads to the register files wherein in a single thread mode, the circuitry enables the read and write ports for both the first register file and second register file for access by a single thread.

[Column 5, lines 21-42 and figure 5C discuss an embodiment with the use of privately used architectural registers for each thread. This means that a first thread has a first set of architectural registers and the second thread has a second set of architectural registers. Column 10, lines 18-41 show that this third embodiment is called SSASR and provides further details of it. Column 15, lines 28-41 show that the architectural registers are set up in eight separate register files, one for each of eight threads (column 13, line 1). Therefore the private architectural register sets are in fact architecture register files. Column 5, lines 21-42 show that an embodiment has private architectural registers for each thread in a simultaneous multithreaded processor (SMT, column 4, line 66 – column 5, line 1) for a multi-thread mode, when the maximum number of threads are executing, where there is no register sharing. This section also shows that at some points in time, the processor will be executing fewer threads than the number of private register sets contained in the processor. This number of threads fewer than the maximum number of threads (eight) includes the case of when a single thread is executing, or a single thread mode. Further proof of this is shown in figures 6A-6D, for example, where it is shown that SSASR implementations (different only by register length) all execute a single thread at times. Column 5, lines 21-42 show that in the SSASR embodiment when fewer threads than the maximum are executed, and thus the single thread mode as well, the architectural registers sets of the non-executing threads or contexts are shared with the active threads. Therefore, when there is a single executing

thread (single thread mode) architectural registers will be used from the second thread, which is idle, and thus in a single thread mode the active thread has access to both the first and second register sets. As shown above when there is one thread active the processor is in single thread mode and when all threads are active the processor is in multi-thread mode, and thus there must be a mechanism to switch the register access properties of the processor for switching between these modes. This mechanism is element 28 of figure 1, which is described in column 7, lines 18-22 and column 10, lines 34-37 as deciding what registers to use for holding data for each thread. Since each thread operates independently for execution they each have their own execution unit banks. Since the register files are accessible by each thread in single thread mode depending on which thread is active, there inherently are read and write ports to each register file for each thread and the execution unit banks therein. There also is inherently circuitry for enabling and disabling the read/write ports of the register files so that when in single thread mode the active thread can use each register file and when in multi-thread mode each thread only has access to its own register file.]

b. Levy does not explicitly disclose wherein a single register name is renamed to refer to a first location in the first register file and a first location in the second register file. However, Levy does disclose the use of register renaming for the SSASR embodiment given above (figure 5C for example) for multiple functional units (figures 1, 3, and 4).

c. Cook has disclosed in figure 5 and column 4, lines 37-68 the idea of broadcasting the data to be written to each register file to all the shared register files to create identical register files or in effect a multi-port register file.

d. In column 5, lines 21-42 along with the background, for example, Levy has recognized that unused portions of the processor when less than the maximum number of threads are executed (including the single thread mode) is a waste of resources and thus proposes sharing these otherwise thread-private resources. Specifically Levy has taught this idea for the use of sharing private register files for renaming, but one of ordinary skill in the art would also have recognized the waste of resources the idle execution units provide and would have been motivated to use all of the execution units of Levy to optimize execution of the single thread in a superscalar manner when a single thread is executing in order to boost performance of that single thread. Column 2, lines 29-40 of Cook along with the section of Cook cited above show that by broadcasting the data to each private yet shared register file, in effect, a fast multi-port type register file is created that allows for quick data transfers to and from the private register file to each corresponding execution unit. This quick performance would have motivated one of ordinary skill in the art to modify the design of Levy to broadcast the writes to each shared register file, when in single thread mode where the resources are shared.

It would have been obvious to one of ordinary skill in the art to modify the design of Levy to use idle execution units in a superscalar manner when in a single thread mode

and to broadcast writes to each register file associated with an execution unit in order to keep identical copies so that increased performance is realized in the system both through superscalar execution and a multi-port nearby register file when a single thread is executed.

22. In regard to claim 18, Levy discloses the microprocessor of claim 16, wherein, in a multi-thread mode, the circuitry enables the read and write ports associated with the first bank of execution units for the first register file for access by a first thread and disables the read and write ports associated with the second bank of execution units for the first register file so as to not be accessible by the first thread, as shown above.

23. In regard to claim 19, Levy discloses the microprocessor of claim 18, wherein, in the multi-thread mode, the circuitry enables the read and write ports associated with the second bank of execution units for the second register file for access by a second thread and disables the read and write ports associated with the first bank of execution units for the first register file so as to not be accessible by the second thread, as shown above.

24. In regard to claim 20, Levy discloses the microprocessor of claim 16 further comprising a register renamer for renaming logical references to registers in instructions so as to operate in one of a single thread mode or multi-thread mode of operation. [As shown above, the processor uses register renaming and thus inherently comprises a register renamer for renaming logical references to registers in instructions and based on the registers available for renaming the processor may be found to be in a single or multi-thread mode.]

25. Claims 4, 5, and 14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Levy in view of Cook and Sollars (5,900,025).

26. In regard to claim 4,

a. Levy discloses the microprocessor of claim 3,

b. Levy does not explicitly disclose wherein the mechanism includes control registers that control whether the microprocessor executes in the single thread mode or in the multi-thread mode.

c. Sollars has taught in column 1, lines 12-31 the use of a number of control registers for controlling hardware and operation modes.

d. Sollars has also taught in this section that all computer systems have such control registers for controlling system operations. The fact that all computer systems have control registers would have motivated one of ordinary skill in the art to modify the computer system design of Levy to use control registers for controlling hardware and operation modes. With these control registers in place, the design of Levy in view of Sollars would have controlled whether operation was in the single thread or multi-thread mode.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Levy to incorporate control registers for controlling hardware and operation modes of the processor as taught by Sollars since all computer systems include control registers for such a purpose.

27. In regard to claim 5, Levy in view of Sollars discloses the microprocessor of claim 4, wherein the mechanism includes software that writes appropriate values into the

control registers to switch between the single thread mode and the multi-thread mode. Column 1, lines 40-55 of Sollars have shown that the control registers are modified by instructions and thus by software. Therefore, software is used to write values into the control registers and control the thread mode.

28. In regard to claim 14,

a. Levy discloses the method of claim 11,

b. Levy does not disclose wherein the microprocessor includes control registers and wherein the method further comprises prompting the switching by writing values in the control registers.

c. Sollars has taught in column 1, lines 12-31 the use of a number of control registers for controlling hardware and operation modes.

d. Sollars has also taught in this section that all computer systems have such control registers for controlling system operations. The fact that all computer systems have control registers would have motivated one of ordinary skill in the art to modify the computer system design of Levy to use control registers for controlling hardware and operation modes. With these control registers in place, the design of Levy in view of Sollars would have controlled whether operation was in the single thread or multi-thread mode.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Levy to incorporate control registers for controlling hardware and operation modes of the processor as taught by Sollars since all computer systems include control registers for such a purpose.

Response to Arguments

29. Applicant's arguments with respect to claims 1-8 and 11-20 have been considered but are moot in view of the new ground(s) of rejection.

30. Applicant's arguments filed 9/15/04 with respect to claims 9 and 10 have been fully considered but they are not persuasive.

31. Applicant has argued regarding claim 9 that Levy does not disclose mapping references to registers in instructions to both the first register file and the second register file when in a single thread mode. The Examiner notes that the claim language does not require in the single thread mode the mapping of a single register name to a register in both the first and second register files, but instead simply states mapping references to registers in both the first and second register file with no mention of mapping to a single name to both registers as given in the other independent claims. Thus, Levy does anticipate claim 9 since he uses register files of inactive threads (including when there is only one active thread – a single thread mode) for extra physical registers (as indicated by Applicant). Column 5, lines 21-42 and figure 5C discuss an embodiment with the use of privately used architectural registers for each thread. This means that a first thread has a first set of architectural registers and the second thread has a second set of architectural registers. Column 10, lines 18-41 show that this third embodiment is called SSASR and provides further details of it. Column 15, lines 28-41 show that the architectural registers are set up in eight separate register files, one for each of eight threads (column 13, line 1). Therefore the private architectural register sets are in fact architecture register files. Column 5, lines 21-42

show that an embodiment has private architectural registers for each thread in a simultaneous multithreaded processor (SMT, column 4, line 66 – column 5, line 1) for a multi-thread mode, when the maximum number of threads are executing, where there is no register sharing. This section also shows that at some points in time, the processor will be executing fewer threads than the number of private register sets (files) contained in the processor. This number of threads fewer than the maximum number of threads (eight) includes the case of when a single thread is executing, or a single thread mode. Further proof of this is shown in figures 6A-6D, for example, where it is shown that SSASR implementations (different only by register length) all execute a single thread at times. Column 5, lines 21-42 show that in the SSASR embodiment when fewer threads than the maximum are executed, and thus the single thread mode as well, the architectural registers (and thus register files) of the non-executing threads or contexts are shared with the active threads. Therefore, when there is a single executing thread (single thread mode) architectural registers will be used from the second thread, which is idle, and thus in a single thread mode the active thread has access to both the first and second register files. As shown above the register files are architectural and thus the instructions register references are directly mapped to the registers in these files. For the multi-thread mode, each thread only access registers private to that thread (first thread/first register file) and for the single thread mode both first and second register files are used with the second register file being used as rename registers for mapping of the architectural registers of the first register file. Column 5, lines 21-42 show that architectural registers files from the idle threads (i.e. second register file from the

second thread) are used for renaming registers and thus are mapped to the architectural registers of the first register file.

Conclusion

32. Applicant's amendment necessitated the new ground(s) of rejection for claims 1-8 and 11-20 presented in this Office action. The arguments for claims 9-10 were not found to be persuasive. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

33. The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Shane F Gerstl whose telephone number is (571) 272-4166. The examiner can normally be reached on M-F 6:45-4:15 (First Friday Off).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Shane F Gerstl
Examiner
Art Unit 2183

SFG
December 16, 2004